A. A Very Short Number Sequence

```
Time limit:   1 second
Memory limit:  65535 kBytes
```

Description

Your task is to answer one of the four questions below:

Question 1

Given k, a natural number ($k \geq 1$), define $A(k)$ as the largest n, for which the list $L = 1, 2, 3, ..., n$ has a sublist K of length k, such that the list $L \setminus K$ contains no Arithmetic progression of length k (sublists of length 1 or 2 by default are considered arithmetic progressions of length 1 and 2 respectively).

- Compute $A(k)$ for $k = 1, 2, 3, 4, 5, 7$.

Question 2

Define $B(k)$ as the maximal number of 1's that a $k \times k$ invertible matrix - containing only 0 or 1 - can have ($k \geq 1$).

- Compute $B(k)$ for $k = 1, 2, 3, 4, 5, 7$.

Question 3

Let $C(k)$ be the maximal number of interior regions formed by k intersecting circles, for ($k \geq 1$).

- Compute $C(k)$ for $k = 1, 2, 3, 4, 5, 7$.

Question 4

Let $D(k)$ be the number of walks of length 3 between any two distinct vertices of the complete graph K_{k+1}, ($k \geq 1$). Example: $D(2) = 3$ because in the complete graph ABC we have the following walks of length 3 between A and B: $ABAB$, $ACAB$ and $ABCB$.

- Compute $D(k)$ for $k = 1, 2, 3, 4, 5, 7$.

Input

There is no input for this problem.

Output

The output should contain six numbers separated each other by a single colon. *Eg.:* 0,1,0,0,0,1

B. Coin Game

Time limit: 1 second
Memory limit: 65535 kBytes

Description

Adam and Bob are brothers who enjoy playing various games with their collection of shiny coins. Right now, they are playing a game in which each kid has one or more **piles** of coins in front of them on the table. Each pile of coins contains some (at least 1) coin pieces. Meanwhile, Dad prepares to go to the post office and send a holiday card to the grandmother of the boys. Sending the postcard requires D coins, but, unfortunately, Dad does not have any coins at hand. He decides to take some of the coins from the kids.

To make it fair, he comes up with the following process to collect the coins: in each turn, he selects one of the piles on the table with equal probability. He then takes one coin from the selected pile, and the turn ends. The process ends when there arent any coins left on the table, or once Dad manages to collect D coins total (i.e., after taking D turns). If a pile does not contain any coin at the end of a turn, then it wont be considered again for selection. Adam and Bob are worried about their precious coins. They want to know the probability of Dad taking all their own coins (the coins in front of Adam are considered to be Adams and vice versa). Help them and compute this probability for both children!

Input

The first line of the input contains N, M, and D, the number of piles in front of Adam and Bob, respectively ($1 \leq N, M \leq 5$), and the number of coins Dad wants to take ($1 \leq D \leq 100$). The second line contains N integers, the number of coins in each pile in front of Adam ($1 \leq A_i \leq 6$). The third line contains M integers, the number of coins in each pile in front of Bob ($1 \leq B_i \leq 6$).

Output

Output two lines. The first line should contain the probability of taking every coin from Adam by the end of the process. The second line should contain the same probability for Bob. Print each probability rounded to 6 digits after the decimal point, in the format X.XXXXXX.

Example

Input	Output
1 2 2 2 1 1	0.111111 0.333333

Explanation: In this example, there is one pile containing two coins in front of Adam, and two piles containing a single coin each in front of Bob. Dad wants to take 2 coins. The only way to take every coin from Adam is to pick the pile in front of him twice, with probability $1/3 \cdot 1/3 = 1/9$.

On the other hand, the only way to take every coin from Bob is to pick one of his two piles first (with probability 2/3). The chosen pile is removed, so the probability of selecting the remaining pile of Bob the second time is 1/2 (as there is one pile in front of each of them at this stage). This gives the probability $2/3 \cdot 1/2 = 1/3$.

Input	Output
2 3 12 3 2 4 2 3	0.565991 0.137738

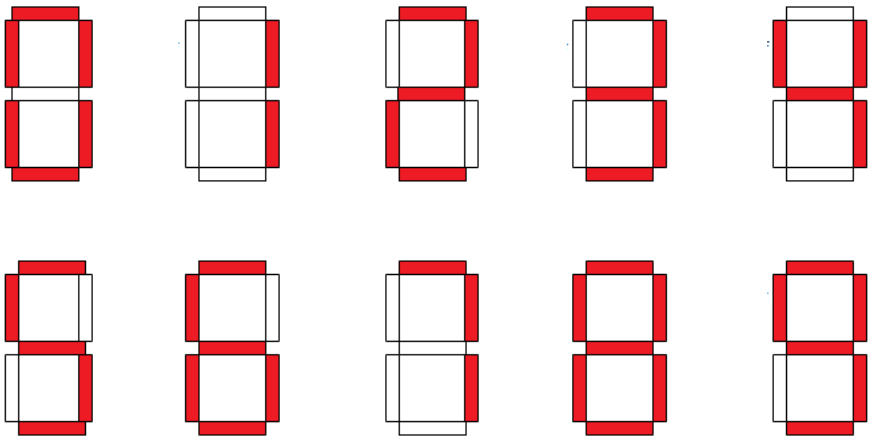
Input	Output
5 4 33 4 5 4 2 6 1 3 6 2	1.000000 1.000000

C. Clocks

Time limit: 1 second
Memory limit: 65535 kBytes

Description

Given any arbitrary time displayed on a digital clock. The clock uses four LEDs, each made up of seven segments, and shows the time in a 24-hour format from 0:00 to 23:59. What is today's closest time to the given time that would require the same number of LED segments to be lit?



Input

The first line of the input contains H, the given hour, and the second line contains M, the given minute.

Output

The output should contain one line:

- the text *Impossible*, if there is no solution,
- OR the solution in *H:MM* format.

Constraints

- $0 \leq H \leq 23$
- $0 \leq M \leq 59$

Examples

Input	Output
1 12	1:13

Input	Output
17 14	17:41

Input	Output
18 58	19:08

D. Döme's party

Time limit: 1 second
Memory limit: 65535 kBytes

Description

During the summer holidays, Döme is organizing a party where he wants to provide burgers for his guests, one for each guest. He can make two kinds of burgers, where each burger consists of a bun, a meat patty and a slice of cheese, but the ingredients are different and not interchangeable. Döme has all the ingredients at home and his mom even gave him money (copper coins) to organize the party. Döme's goal is to invite as many guests as possible, which means he needs to be able to make as many burgers as possible. Help him solve his problem.

Input

- The first line of the input contains the number of buns (b_1) he has at home and the price of the buns (pb_1) for the first burger. The second line contains the number of meat patties (m_1) and their price (pm_1), the third line contains the number of cheese slices (c_1) and their price (pc_1).
- The next 3 rows contain the same values for the other type of hamburger (b_2, pb_2, m_2, pm_2, c_2 and pc_2).
- The last row has a single number, the amount of money he received for the party (m).

Output

The output is a single number that tells you the maximum number of guests Döme can invite, i.e. the maximum number of burgers he can cook.

Constraints

- $0 \leq b_1, m_1, c_1, b_2, m_2, c_2 \leq 10^6$
- $1 \leq pb_1, pm_1, pc_1, pb_2, pm_2, pc_2 \leq 10^6$
- $1 \leq m \leq 10^6$

Example

Input	Output
2 5 3 7 3 3 2 8 1 2 1 1 20	6

Explanation: Döme buys a bun for his first hamburger (5 coppers) so he can make 3 of the first kind. He then buys a bun (8 coppers), 2 meat patties (4 coppers) and 2 slices of cheese (2 coppers) for the second one, so he can make 3 of those too, and has used 19 coppers in total.

E. Alibaba and the Magical Cave of Whispering Lamps

Time limit: 1 second
Memory limit: 65535 kBytes

Description

In the sun-scorched deserts of ancient Persia, where mirages dance on the horizon and djinns whisper secrets to the wind, Alibaba and his band of clever thieves stumbled upon a treasure beyond imagination. Gold coins that shimmered like starlight, jewels that held the essence of rainbows, and artifacts of power long forgotten by time itself. Knowing the dangers that come with such wealth, Alibaba devised a plan to protect their fortune. They concealed the treasure in an unbreakable vault, secured by n magical locks (where n is the number of thieves, between 2 and 40). Each lock could only be opened by a key inscribed with a thief's true name, ensuring that all must be present to claim their prize. This way, no single thief or partial group could retrieve the treasure alone. But the cunning Grand Vizier, his eyes gleaming with greed, learned of their scheme through his network of spies. Swift as a desert viper, he struck in the dead of night, pilfering the thieves' keys while they slumbered. The Grand Vizier then hid the keys within n enchanted lamps, deep within the Cave of Whispers, a place of ancient magic.

The Challenge of the Whispering Lamps

After a perilous quest across blistering dunes and treacherous mountains, Alibaba and his band finally located the hidden oasis containing the Cave of Whispers. Inside, there were the n magical lamps arrayed in a perfect line, their surfaces gleaming with ethereal light. A mischievous djinn, amused by their plight, appeared in a swirl of purple smoke. It taught them a magical chant to activate the lamps, but warned of the cave's fickle magic:

- The thieves must enter one by one, unable to share information with those who came before or after. They must coordinate their strategy before the first thief enters the cave.
- Each thief may examine up to k lamps before the cave's magic drains them, whisking them away to the Grand Vizier's Dungeon of Eternal Night and Despair.
- The lamps cannot be touched, marked, or manipulated in any way other than through the magical chant.
- Should any thief fail to recover their key, the treasure would remain locked forever, as all n keys are required to open the vault.

As the djinn prepared to vanish in a swirl of shimmering smoke, its voice resonated in a riddle-like verse.

"By the names you carry and the order they bind,
Begin your search where they first align.
Where your path starts, let your discovery flow,
For the cycles of fate favor those who know."

The Magical Chants

To activate a lamp, a thief must recite the following:

"O lamp within the cycle, radiant and bold,
Illuminate the hidden path that you hold!
I, [*Thief's Name*], seek the key or the next name in line,
Unveil the magic, let our fates entwine!"

The lamp then responds in one of two ways:

1. If it contains the thief's key:

   "Seeker true, your search is done,
   The cycle ended; your key is won!
   [*Thief's Name*], claim what's yours by right,
   And to the treasure, take your flight!"

   The lamp then ejects the key and teleport the thief to the vault of treasures.

2. If it contains or once held another thief's key:

   "Not for you, this lamp's embrace,
   You sought in [*Other Thief's Name*]'s place.
   Continue the cycle, persistent one,
   Your journey is not yet done."

Alibaba, the clever leader of the thieves, listens closely to the djinn's parting riddle and the magical chant and possible responses, gradually unraveling their hidden meaning. From these cryptic clues, he devises a brilliant strategy to greatly enhance the chances that every thief will retrieve their key. Can you also decipher the strategy that will maximize their chances of success?

Alibaba's Conundrum

The djinn has revealed that each thief can examine only a fixed number of lamps k before the cave's magic takes effect. However, Alibaba is not able to exactly quantify the risk by its own. What Alibaba does know is his own risk tolerance. He has determined a minimum probability of success p that he is willing to accept for this dangerous venture. Now, he must figure out: what is the minimum number of lamps (let us call it k_{min}) that each thief needs to be able to examine for their collective chance of retrieving all keys and the treasure to be at least p? If the djinn's revealed k is greater than or equal to this k_{min}, Alibaba will assume their chances and lead his band into the cave. If not, they will need to retreat and return another day, perhaps with magical amulets that

could increase their resistance to the cave's draining magic, effectively allowing them to examine more lamps.

Your Quest

Brave codesmith! Alibaba turns to you in this crucial moment. Your task is to calculate k_{min}, given:

1. The number of thieves (n)
2. Alibaba's desired minimum probability of success (p)

Remember, the fate of the treasure - and the freedom of the thieves - rests upon your calculations!

Input

One line with two numbers:

- An integer n: The number of thieves (and lamps)
- A real number p, specified with a precision of up to 10^{-12}: Alibaba's desired probability threshold

Output

Your program must print to the standard output the k_{min} value.

Constraints

- $(2 \leq n \leq 40)$
- $(0 < p \leq 1)$

Example

Input	Output
2 0.000001	1

Input	Output
10 0.000001	2

Input	Output
13 1.0	13

Input	Output
40 0.8	33

Notes

- The actual probability of success may exceed the given threshold.

- Each thief's name is unique.
- The keys are randomly distributed among the n lamps, each lamp containing one key.
- All lamps always appear identical, regardless of whether they still hold a key or have been already successfully searched.
- Alibaba does not possess a key, nor does he enter the cave himself. He serves solely as the strategist and leader.

F. Treasure Map

Time limit: 1 second
Memory limit: 65535 kBytes

Description

The Treasure Map

In the quiet village of Numeria, an old cartographer is famous for his intricate maps and mysterious puzzles. Before retiring, he created his final treasure map, which, according to rumors, leads to a hidden trove of ancient coins. The map is unique. Instead of typical landmarks, it contains a grid of numbers. Each position in the grid holds a value that serves as a clue to locating the treasure. The villagers soon realized that to decode the map, they would need to apply a specific operation XOR on the coordinates. To determine the exact location of the treasure, they must find the K-th largest XOR value among all the grid's coordinates. This value is essential as it directly points to the location of the hidden treasure.

The Challenge

Help the villagers locate the treasure by writing a program to determine the K-th largest XOR coordinate value. Given an $n \times m$ matrix of natural numbers and a number K, the program must compute the XOR for each coordinate and return the K-th largest value (1-indexed, meaning there is a first largest, not a "0-th" largest). The value at the matrix coordinate (a, b) is the XOR of all matrix elements (i, j) where $0 \leq i \leq a < m$ and $0 \leq j \leq b < n$ (indexes starting from 0).

The XOR (exclusive OR) operation is a logical operation that compares corresponding bits of two numbers and produces a result according to the following rule:

- The result bit is 1 if the bits being compared are different.
- The result bit is 0 if the bits being compared are the same.

Input

The first line contains the values n, m and K. The remaining n lines contain the rows of the matrix, with m elements on each line.

Output

The output should contain a single number, the K-th largest XOR coordinate value.

Constraints

- $1 \leq n, m \leq 1000$
- $0 \leq matrix[i][j] \leq 10^6$
- $1 \leq K \leq m * n$

Example

Input	Output
2 2 1 5 2 1 6	7

Explanation: The result is 7, because that is the largest XOR value (the value at coordinate $(0, 1)$ is calculated as 5 XOR 2 = 7).

Input	Output
2 2 4 5 2 1 6	0

Explanation: The value at coordinate $(1, 1)$ is calculated as 5 XOR 2 XOR 1 XOR 6 = 0, which is the 4th largest value.

G. Flood Resistance

Time limit: 1 second
Memory limit: 65535 kBytes

Description

Floods have become a recurring problem in Hungary in recent years. Even though most cities are well protected, parts of the road network are still exposed to occasional floods, which might make it impossible to reach certain parts of the country.

In this problem, we model the road network of Hungary as N cities connected by M bidirectional roads. Each road connects two different cities a_i and b_i, and it is possible to reach any city from any other city by traversing one or more roads. Furthermore, we know the height of the lowest point of each road, represented by an integer h_i: this means that the lowest point of road i lies h_i millimeters above sea level. During a flood, if the water level rises above h_i millimeters, the road becomes impossible to traverse and is closed. The road is **safe** to use whenever the water level is less than or equal to h_i.

The government established a call center to help people traveling the country during a flood. Citizens can ask the center whether city y is **reachable** from city x at the current time. Suppose that the water level is l millimeters above sea level at the time. We say that city x is reachable from city y if and only if it is possible to get from city x to city y using only safe roads when the water level is l.

Your task is to write a program that, given the description of the road network, computes the answer to Q such calls.

Input

The first line of the input contains the integers N, M, and Q, the number of cities ($1 \leq N \leq 100000$), the number of roads ($0 \leq M \leq 100000$), and the number of calls ($1 \leq Q \leq 100000$), respectively.

Each of the next M lines describes a road. The i-th line contains three integers a_i, b_i, and h_i ($1 \leq a_i, b_i \leq N, 0 \leq h_i \leq 10^9$), denoting the two cities connected by the road and the height of its lowest point.

The last Q lines describe the calls made by citizens in an **online** manner: you must answer a call first before learning the parameters of the next call. Let the j-th line contain the integers x'_j, y_j', and l'_j, and suppose that there were z calls so far to which the correct answer was YES. Then, the parameters of the j-th call are $x_j = x'_j \bigoplus z$, $y_j = y'_j \bigoplus z$, and $l_j = l'_j \bigoplus z$ ($1 \leq x_j, y_j \leq N, 0 \leq l_j \leq 10^9$), asking whether city y_j is reachable from city x_j when the water level is l_j. Here, the symbol $\bigoplus$ denotes the bitwise XOR operator.

Output

You must print Q lines, the j-th of which contains the answer to the j-th call: YES, if we can reach one city from the other at the current water level, and otherwise NO.

Example

Input	Output
4 4 3 1 2 9 2 3 5 3 4 4 4 1 2 2 4 6 1 4 4 0 2 4	NO YES YES

H. Ultrastar

Time limit: 1 seconds
Memory limit: 65535 kBytes

Description

John Doe is planning a party again, but this time it will be a karaoke party! To make things more fun, he wants to use UltraStar, an open-source karaoke game which scores the singers automatically.

John downloads n songs from different places on the internet, he loads all of them into the game, starts up the software, and... surprise, it hangs! It looks like there are k bad songs among the n, and if John loads even one of them into the game, it won't start. He only has t minutes until the party and it takes exactly one minute to start up the game with a certain subset of the n songs. Help John in making the party a success, by determining all of the bad songs!

Input and output

This is an *interactive problem*, which means that you should not read the input and write the output in the usual way, rather you have to interact with the grader program by following these steps:

1. Read from the standard input the values of n, k, and t.
2. In order to start up the game with a certain subset of songs, write to the standard output a line having the format `1` m $i_1, i_2, \ldots, i_m$ (with all numbers separated by a single space), where m is the number of songs in the subset and $i_1, \ldots, i_m$ are the indices of the songs in any order. It's important to flush the output buffer after this (eg. by using `endl` in C++ or `flush` in Pascal).
3. After each query you should read a single number from the standard input, then proceed to step 2 or step 4. The number will be 0 if the game doesn't start up, and 1 if it does.
4. In order to give a final answer write to the standard output a line having the format `2` $j_1, j_2, \ldots, j_k$ (with all numbers separated by a single space), where $j_1, \ldots j_k$ are the indices of the bad songs.

Constraints

- $1 \leq n \leq 1000$
- $1 \leq k \leq n$
- $0 \leq t \leq 100\ 000$

- $1 \leq m \leq n$ otherwise you will receive a *Wrong Answer* judgement.
- $i_1, \ldots, i_m$ should be all distinct, otherwise you will receive a *Wrong Answer* judgement.
- $j_1, \ldots, j_k$ should be all distinct, otherwise you will receive a *Wrong Answer* judgement.
- If you start up the game more than t times, the grader will stop the execution of your program and you will receive a *Wrong Answer* judgement.
- The songs are numbered from 1 to n, if you output an index outside of the $[1, n]$ interval you will receive a *Wrong Answer* judgement.
- Your solution will be evaluated on 20 test cases having the following values:

Case	n	k	t
1	4	2	1
2	5	2	5
3	6	3	5
4	1	1	0
5	1000	20	226
6	3	3	0
7	900	30	289
8	998	499	1
9	999	499	1
10	997	499	1004
11	999	500	999
12	900	300	1448
13	800	100	647
14	1000	250	2
15	999	998	1747
16	1000	200	1137
17	1000	908	1978
18	1000	100	736
19	1000	10	36
20	100	40	98

- *For each test case* it is guaranteed that there is a deterministic (non-randomized) strategy that solves the problem using no more than t queries. *Hint:* You may need to use a different strategy for ***odd*** and **even** cases, respectively.

Example

If $n = 4, k = 2$, and the bad songs are the ones with indices 3 and 4, a possible interaction:

Standard output	Standard Input
1 2 1 2 2 3 4	4 2 1 1

I. Casino

Time limit: 1 second
Memory limit: 65535 kBytes

Description

A new game called "Double or Nothing" has appeared in the casinos of Las Vegas. The game is played by a single player against the computer, which acts as the dealer during the game. The simplified rules of the game are as follows: each time, the player decides the stake and places a chosen amount of money in the center of the table. The dealer matches this amount with an equal bet in the center. The dealer then decides whether to grant the entire amount on the table to the player or to keep it for themselves. The following rules apply to the game:

- The player initially has a certain amount of money, V, in dollars ($), which is entered into the game program as a virtual amount at the beginning.
- The player can only bet as much money as they currently possess. This amount changes after each round, sometimes decreasing, sometimes increasing. As a result, the stake value can vary in each round, potentially reaching the full amount or even zero.
- The game consists of $d + z$ rounds, meaning the dealer is obligated to double the amount d times (grant the amount on the table to the player) and zero out the amount z times (keep the amount for themselves).
- The dealer always plays optimally, making decisions to maximize their winnings by the end of the game, as long as they can choose from both types of actions.
- The stake can be a fractional amount. In such cases, the value bet on the table is represented by two natural numbers a and b, with the stake value expressed as the irreducible fraction a/b.

Task Requirements

Given the values of V, d, and z for n games, calculate the final amount in the form of an irreducible fraction u/v (where u and v are natural numbers that are relatively prime) for each V, d, z triple, representing the player's final winnings. Definitions:

- n input value, the number of games.
  For each game:
- V input value, the player's starting amount.
- d input value, the number of rounds in which the player wins the amount.

- z input value, the number of rounds in which the player loses the amount.
- u v output values, where the fraction u/v represents the final amount won by the player.

Input

The first line contains the value of n, the number of games. The following n lines each contain three natural numbers V, d, and z, separated by spaces, as described above.

Output

The output will contain n lines. Each line contains two natural numbers u and v, separated by a space, which represent the irreducible fraction u/v for the maximum winnings achievable by the player in each game.

Constraints

- $0 \leq V \leq 1000$, a natural number
- $0 < z, d, n \leq 15$, natural numbers
- The result always fits within the 64 bit integer type.

Example

Input	Output
4	500 1
500 0 1	1246 1
623 1 0	400 1
300 1 1	1196 3
299 1 1	

Explanation: n=4, so four games will be played.

First game: The player starts with \$500. The dealer has $d = 0$ doubles and $z = 1$ chances to keep the amount.

In this case, the player plays with a \$0 stake, which the dealer keeps, so the player loses nothing and retains \$500, written as the irreducible fraction 500/1, so the output will be: 500 1.

Second game: The player starts with \$623. The dealer has $d = 1$ double and $z = 0$ chances to keep the amount.

Here, the player stakes the full \$623, and since the dealer must double, the final amount is \$1246, written as 1246/1, so the output will be: 1246 1.

Third game: The player starts with \$300. The dealer has $d = 1$ double and $z = 1$ chances to keep the amount.

The player's first stake is \$100. If the dealer returns it, the player gains \$100 and has \$400. Now the dealer has only one z move left, and the player stakes \$0 in the second round, which the dealer keeps, leaving the player with \$400.

Alternatively, if the dealer keeps the \$100 initially, the player is left with \$200. In the second round, the dealer doubles the player's full amount, returning \$400. Either way, the player ends with \$400, written as 400/1, so the output will be: 400 1.

Fourth game: the player's starting amount is \$299. The dealer has $d = 1$ doubling option and $z = 1$ zeroing option. The player's first bet will be 299/3, or approximately \$99.67.

If the dealer returns it, the player wins \$99.67 and now has 299+299/3=1196/3, or approximately \$398.67. The dealer now has only one zeroing option left. In the second round, the player bets \$0, which the dealer must keep, leaving the player with \$398.67.

If, instead, the dealer keeps the \$99.67 in the first round, the player is left with approximately \$199.33. The dealer then has only one doubling option remaining. In the second round, the player bets their full amount, doubling it to get an additional \$199.33, resulting in a final total of \$398.67.

Therefore, regardless of the dealer's moves, the player will finish the game with \$398.67 (written in irreducible fraction form as 398.67=1196/3), so the output will be: 1196 3.

If in the first round the player were to risk \$99, the dealer would choose the doubling operation, and the player would end up with \$398. In the second round, the player would not bet anything and would keep the \$398, which is less than in the case of optimal play.

If in the first round the player were to risk \$100, the dealer would keep the amount, and the player would be left with \$199. In the second round, the player would bet the entire amount, which would double, resulting in a final profit of 2 * \$199 = \$398, which is also less than in the case of optimal play.

J. Climbers

Time limit: 3 seconds
Memory limit: 128 kBytes

Description

A group of ECN competitors decides to go hiking along a mountain ridge.

A mountain sequence is a sequence of numbers that meets the following criteria:

- Unique Peak: The sequence has exactly one peak element, which is strictly greater than its immediate neighbors (if neighbors exist).
- Monotonic Increase Before the Peak: All elements to the left of the peak (if any exist) form a monotonically increasing sequence, meaning each element is greater than or equal to the preceding one.
- Monotonic Decrease After the Peak: All elements to the right of the peak (if any exist) form a monotonically decreasing sequence, meaning each element is less than or equal to the preceding one.

A square matrix is said to have a mountain ridge if the following conditions are satisfied:

- Each row of the matrix forms a valid mountain sequence, meaning each row contains a unique peak element that satisfies the mountain sequence criteria.
- The peaks of consecutive rows are neighboring elements, meaning the peak in one row is adjacent (vertically, or diagonally) to the peak in the next row.

Task

The program should:

- Read a natural number n and an $n \times n$ matrix of natural numbers from the standard input.
- Print `YES` to the standard output if the matrix contains a valid mountain ridge.
- Print `NO` to the standard output if no valid mountain ridge exists.

Input

The input format is as follows:

- The first line contains a single natural number n.
- The next n lines contain n space-separated natural numbers, representing the elements of the $n \times n$ matrix.

Output

Output a single line:

- Print `YES` if the given matrix forms a valid mountain ridge.
- Print `NO` otherwise.

Constraints

- $2 \leq n \leq 2500$
- $0 \leq matrix[i][j] \leq 10^9$
- Remember: the *memory limit* refers to all included libraries as well as to every declared variable. So be careful, it is not as much as it seems!

Example

Input	Output
7 1 2 3 **4** 3 2 1 1 3 4 7 **8** 4 1 3 4 6 7 **8** 2 1 1 2 3 4 5 6 **7** 0 1 2 4 5 **6** 5 3 2 1 5 3 4 4 0 2 8 **9** 7 5 1	`NO`

Explanation: The result is `NO`, because the peaks of the mountain sequences in consecutive rows neither align in the same column nor in adjacent columns. Furthermore, the penultimate row does not form a valid mountain sequence.

Input	Output
5 0 1 1 **4** 2 3 4 5 7 **8** 0 1 2 **8** 3 0 3 **7** 6 6 1 3 **5** 4 3	`YES`

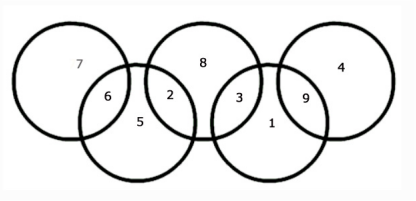
Explanation: The result is `YES`, because the peaks of the mountain sequences in consecutive rows are either in the same column or in adjacent columns, thereby forming a valid mountain ridge.

K. Olympic Rings

Time limit: 1 second
Memory limit: 65535 kBytes

Description

The Olympic rings define 9 closed regions. Place all the numbers $1, 2, 3, 4, 5, 6, 7, 8, 9$ in these regions so that the sum of the numbers in each ring to be the same. The example below shows such an arrangement: 765283194.



List all possible solutions in lexicographic order. Each line of the output file should contain a solution in the same order as in the example (765283194, that is: start, down, down, up, up, down, down, up, up)

Input

There is no input for this problem.

Output

Each line in the output file contains a solution in the same order as in the example. The lines must be in lexicographic order.

L. Recognizing the Language

```
Time limit:   5 seconds
Memory limit: 65535 kBytes
```

Description

Develop a program to recognize in which language is written a certain text, based on some dictionaries.

The standard input consists of the following data:

- n ($2 \leq n \leq 5$) languages / dictionaries;
- each language / dictionary is described by *name*, its *length* (≤ 10000, the number of words), and the actual words;
- after the last language / dictionary, an unspecified number of *phrases* (≤ 15) to be analyzed are following; every phrase is described by its length (number of words ≤ 5000), and the actual words.

Every word, including the language, is a case sensitive string containing up to 15 letters.

Your mission is to identify which language(s) each phrase belongs to. Check, for every word of the phrase, in which dictionaries can be found (some words may be found in two or more dictionaries). Determine, for every language, the number of recognized words, and select the maximum one. If, for a particular language, the number of recognized words exceeds half the length of phrase, display that language. If more than one language is detected (the same number of words were detected), separate the languages by comma and space; display the languages in the order of their appearance (not necessarily alphabetic). If the number of recognized words does not exceed half the length of phrase, display ? (question mark). Do not check if a particular word of any phrase is repeated, manage them as distinct.

Input

See the description.

Output

See the description.

Example

Input	Output
2 rom 4 luna mare ziua miercuri ita 3 luna mezzo mare 4 ziua luna mezzo mare 3 luna mezzo mare 4 patrone merchant luna mare 3 ziua miercuri ombrest	rom, ita ita ? rom

Explanation: The input data includes two languages / dictionaries: rom (4 words) and ita (3 words). You have to analyze four phrases containing 4 words, 3 words, 4 words, and 3 words, respectively.

M. Market

Time limit: 2 seconds
Memory limit: 128 MBytes

Description

Lately, the global economy has experienced positive growth, and all the stocks have increased their price.
Some data collected about the stock market is presented to you as two arrays of integers: C_i-the index of the stock whose price has increased on day i, and V_i- the value by which the price of the stock C_i has increased on day i.

You want to complete the technical analysis of the data collected about the market, to decide whether the market is favorable for investments. To accomplish this, you are presented with a list of Q queries of the form (l_i, r_i). The answer to the i-th query is the greatest increase in value of a stock in the time interval from day l_i to r_i, inclusively.

Input

The first line of input contains two integers, N and Q-the size of the arrays C_i and V_i, and the number of queries, respectively.

The second line of input contains N integers. The i-th integer is C_i, representing the index of the stock whose price has increased on day i.

The third line of input contains N integers, the i-th integer being V_i, the value by which the price of the stock C_i has increased.

Next, Q lines follow. The i-th of these lines contains 2 space separated integers, l_i and r_i, representing the interval for which the result specified in the statement must be calculated.

Output

Print Q integers, each in a separate line representing the answers to the Q given queries.

Constraints

- $N \leq 2 \cdot 10^5$
- $Q \leq 2 \cdot 10^5$
- $1 \leq C_i \leq N$
- $1 \leq V_i \leq 10^9$

Example

Input	Output
10 5 1 1 2 3 2 2 1 1 3 1 9 8 6 9 9 2 8 3 6 1 4 6 2 9 2 6 2 7 3 7	11 19 17 17 17

N. String Simplification

Time limit: 1 second
Memory limit: 65535 kBytes

Description

Let us define the complexity of a string S to be the number of distinct characters in it. For example, the complexity of the string $S = better$ is 4. Dominic doesnt like complicated strings, so he decided to simplify any string he receives. To achieve this, he erases some (possibly 0) characters from the string so that the complexity of the remaining string will be at most 2. Dominic is given a list of T strings. Help him to find for each string the minimum number of characters he must erase to make the string simple enough.

Input

The first line of the input contains T, the number of strings ($1 \leq T \leq 100$). Each of the next T lines contains a string S consisting of the characters $a - z$ (that is, the lowercase letters of the English alphabet). The length of each string is between 2 and 100, inclusive.

Output

Output T lines (one line for each string) containing the minimum number of characters Dominic must erase to make the complexity of the string lesser than or equal to 2.

Example

Input	Output
6	4
string	2
better	0
zzzzz	1
assesses	2
assassins	12
ambidextrously	

Explanation: In the case of $S = string$, after erasing the first four characters, the remaining string is ng, which has a complexity of 2. It can be proven that the complexity cannot be made lesser than 3 by erasing at most 3 characters. In the case of $S = better$, erasing the first and the last character results in the string $ette$, which has a complexity of 2. In the case of $S = zzzzz$, the complexity of S is 1, so there is no need to erase any characters.

O. Birthday Cake

Time limit: 1 second
Memory limit: 65535 kBytes

Description

You're turning 20 on today's contest day. On this occasion, you would like to have a cake made with a formula on it, consisting of the digits of your date of birth, along with some arithmetic operators and parentheses. The value of the formula should equal your age. You've seen this happen several times in your circle of friends, and you've even taken photos of it (see below).

When you think about the task, you realize that such a formula is not so easy to produce. Therefore, you plan to create a program for yourself that will generate all possible formulas for you.

The first step is to turn the digits of your date of birth into meaningful numbers in every possible way.

When you have a sequence of numbers (in your case $2, 0, 0, 4, 1, 12, 3$ is one such sequence), all you have to do is write operators and parentheses between and around the numbers, so that the value of the formula is your age. For example: $(2 + 0 + 0 + 4 - 1) * 12/3 = 20$. If you manage to get this far, you can even have the formula written on your cake. However, it's not always possible to calculate your age from a sequence of numbers, no matter what operators and however many parentheses you try to use.

Your Task

Since generating formulas doesn't seem an easy task, you only need to generate the valid number sequences, but you have to generate them in all possible ways.

A sequence of numbers is considered valid if it contains the digits of your date of birth, and the digits are in the same order as they appear in your date of birth. No more, no less, and no other digit shall appear in a valid sequence of numbers. No number in a sequence can begin with 0, but 0 itself is considered a valid number.

Input

The input contains at most 5 test cases, one per line.

Each test case consists of a date in Hungarian format ($YYYY.MM.DD.$, where $1900 \leq YYYY \leq 2100$, $01 \leq MM \leq 12$, $01 \leq DD \leq 31$).

Output

For each date of birth read from the input, write a block of lines to the standard output. The first line of the block should specify the number of valid number sequences for that date of birth. The remaining lines of the block should contain the valid number sequences themselves, one per line, in lexicographically ascending order. The numbers in each sequence should be separated by exactly one space.

The blocks for two consecutive dates of birth should be separated from each other by exactly one blank line.

Example

Input	Output
1900.01.01.	30 1 9 0 0 0 1 0 1 1 9 0 0 0 10 1 1 9 0 0 0 101 1 90 0 0 1 0 1 1 90 0 0 10 1 1 90 0 0 101 1 900 0 1 0 1 1 900 0 10 1 1 900 0 101 1 9000 1 0 1 1 9000 10 1 1 9000 101 1 90001 0 1 1 900010 1 1 9000101 19 0 0 0 1 0 1 19 0 0 0 10 1 19 0 0 0 101 190 0 0 1 0 1 190 0 0 10 1 190 0 0 101 1900 0 1 0 1 1900 0 10 1 1900 0 101 19000 1 0 1 19000 10 1 19000 101 190001 0 1 1900010 1 19000101

P. Vietnam Training

Time limit: 1 second
Memory limit: 256 MBytes

Description

This is an interactive problem. It's 1971 and the Vietnam War is raging. Nguyen Phan is a vietnamese general and he wants to test a new strategy. His task is to make reconnaissance more efficient on the battlefield, using soldiers and tunnels. There are n soldiers placed at integer coordinates on the battlefield. The soldiers are indexed with integers from 1 to n. The i-th soldier has coordinates (x_i, y_i), which are secret. Two soldiers are visible to each other if there are no other soldiers on the straight segment between them.

With a single command (query), Nguyen Phan can order one of his soldiers to travel through the underground tunnels and pop his head out at coordinates x and y. He can get the following response:

- if at (x, y) there is another soldier with index i, then the soldier that was sent through the tunnels will go back to the base and tell Nguyen Phan the index of the soldier that was already there, $-i$;
- otherwise, the soldier that was sent through the tunnels will STAY at coordinates (x, y) with index t, where t is the total current number of soldiers on the battlefield, including the one sent with this command; the soldier sent returns the list of the indices of all soldiers visible from (x, y).

Nguyen Phans task is to find the coordinates of the initial n soldiers using a limited number of commands! Because you are his assistant, he wants YOU to carry out this task.

Interaction Protocol

The first line contains a single integer $n(1 \leq n \leq 400)$, the number of soldiers. The secret coordinates satisfy $0 \leq x_i, y_i \leq 100$ and all n locations are distinct.

To ask a query, output **?** followed by two integers x and y ($-10^9 \leq x, y \leq 10^9$). If there is already a soldier in (x, y), the result will be $-i$, where i is the index of that soldier. Otherwise the result is given by an integer k, the number of soldiers visible from (x, y), followed by k integers $i_1, ..., i_k$, the indices of the visible soldiers in increasing order ($i_j < i_{j+1}$ for all $1 \leq j < k$). You can use at most 400 queries.

Once you have determined the coordinates of the original soldiers, first output **!** in a single line. Then output n lines, with two integers x, y_i in the i-th line, being the coordinates of the i-th soldier. After that your program should terminate.

Example

Standard input	Standard output
3 2 1 3 3 1 2 4 -5 4 2 3 4 5 -2	 ? 2 3 ? 6 6 ? 6 6 ? 1 11 ? 4 7 ! 3 5 4 7 2 8

Q. Intergalactic Census Sequence Analysis

```
Time limit:   0.5 seconds
Memory limit:  65535 kBytes
```

Description

In response to recent escalations in computational demand, the Intergalactic Census Bureau (ICB) has issued a task of extraordinary technical and strategic importance. This mission, categorized under "High Galactic Priority," aims to enhance population modeling accuracy across interstellar regions by leveraging a numerical dataset with cosmic implications. The specific numeric patterns embedded within this dataset are believed to affect high-energy astrophysical simulations and facilitate projections for interstellar census efforts that span multiple spatial and temporal dimensions.

To proceed, the ICB requires the assistance of a skilled analyst capable of processing an at least one-length sequence, hereinafter referred to as sequence S. This sequence must be parsed and analyzed to support critical functions within the Bureaus overarching framework. While this appears a straightforward extraction of information, its true complexity is deeply tied to the positional, modular, and cumulative properties hidden within the values.

Each element of S is a positive integer of up to six digits, with the exception of a terminating '0' that signifies the end of the sequence. This '0' is not part of the data to be analyzed and should be excluded from all calculations. As an ordered collection of numeric data, the structure of S was crafted under specific cosmic simulation requirements, and each integer within it plays a unique role in contributing to the Bureaus model calibration efforts. The sequence is composed to hold essential positional significance, in that particular values and their relative positions correspond to parameters influencing the ICBs quantum telemetry validations. Additionally, each numeric digit within S is to be considered as carrying intrinsic computational weight, with potential interactions unfolding across positions.

The objective of this task is to conduct an exhaustive analysis of the sequence, identifying the position of a distinct element based on dynamic cumulative conditions that develop as elements in the sequence are processed. Specifically:

1. As you traverse through S, calculate a running cumulative sum of the individual digits across all elements encountered up to and including the current element; This cumulative digit sum forms the basis for the validation check.

2. For each element in S, determine whether this cumulative digit sum equates to the remainder when the current element is divided by the standard intergalactic base year, which is always the current year; This modulo operation is essential, providing an alignment of the data with temporal anchoring crucial for intergalactic time-series analyses.

3. Identify the position (1-based indexing) of the first element that meets the above criteria, which aligns its remainder under division by the base year with the cumulative digit sum obtained up to that point; This position is critical for parameterization within the ICB model.

4. The sequence ends when the value '0' is encountered, and this '0' should not be included in any processing or calculations; In the event that no such element within S meets the specified condition, output the value '-1' to indicate the absence of any qualifying element.

Once you have identified the position of the first element that satisfies the condition, output this position (1-based indexing). If no element satisfies the condition, output '-1'. No additional formatting, whitespace, or extraneous characters should accompany this output, as it will be integrated into the high-precision ICB modeling infrastructure. This precision is vital to ensure compatibility with the Bureau's data ingestion systems, enabling direct impact on the quantum population flux calculations and furthering the Bureau's intergalactic simulation capabilities.

Input

The input contains a single line consisting of positive integers, and a final 0.

Output

The output should contain a single number.

Constraints

- $1 \leq s_i < 1000000$ - elements of the sequence
- There is no information regarding the number of sequence elements.

Example

Input	Output
31 7094 66127 9198 4 6702 645242 9534 720600 227 193079 16883 296 8 781699 269 336895 345 58257 747708 0	18